University of Windsor
Department of Electrical and Computer Engineering
Second M.A.Sc Seminar

Efficient Quadratic Placement for FPGAs

Student: Yonghong Xu
Supervisor: Dr. M.Khalid

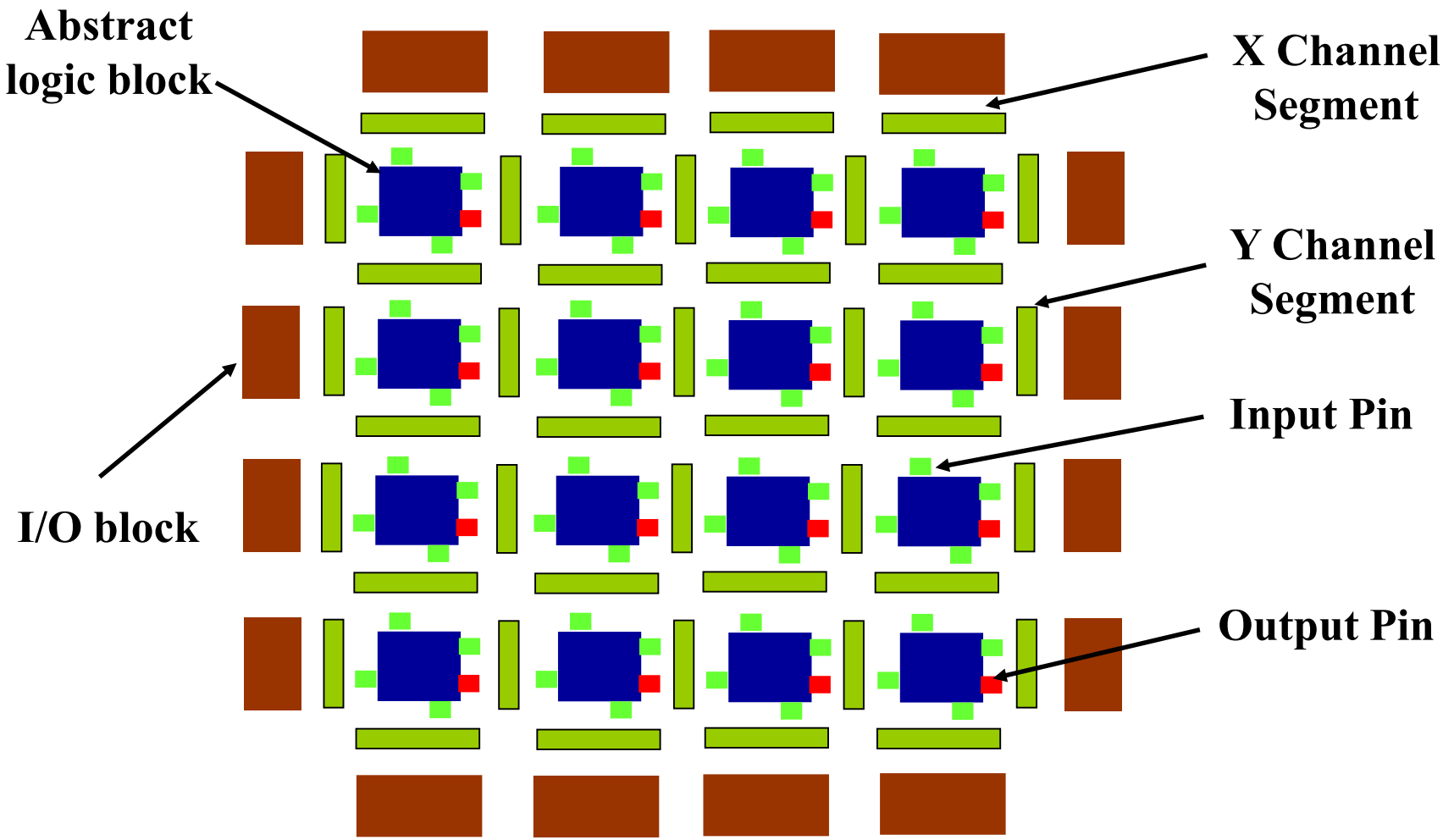
Outline

- Introduction to placement for FPGAs
  - What is an FPGA?
  - What is placement?
  - General placement algorithms
  - Quadratic placement algorithm
- Proposed quadratic placement algorithm
  - Problems with quadratic placement
  - Proposed improvement
  - Details of proposed quadratic placement algorithm
- Experiment results
  - Benchmark circuits
  - Results and analysis
- Contributions
- Questions?

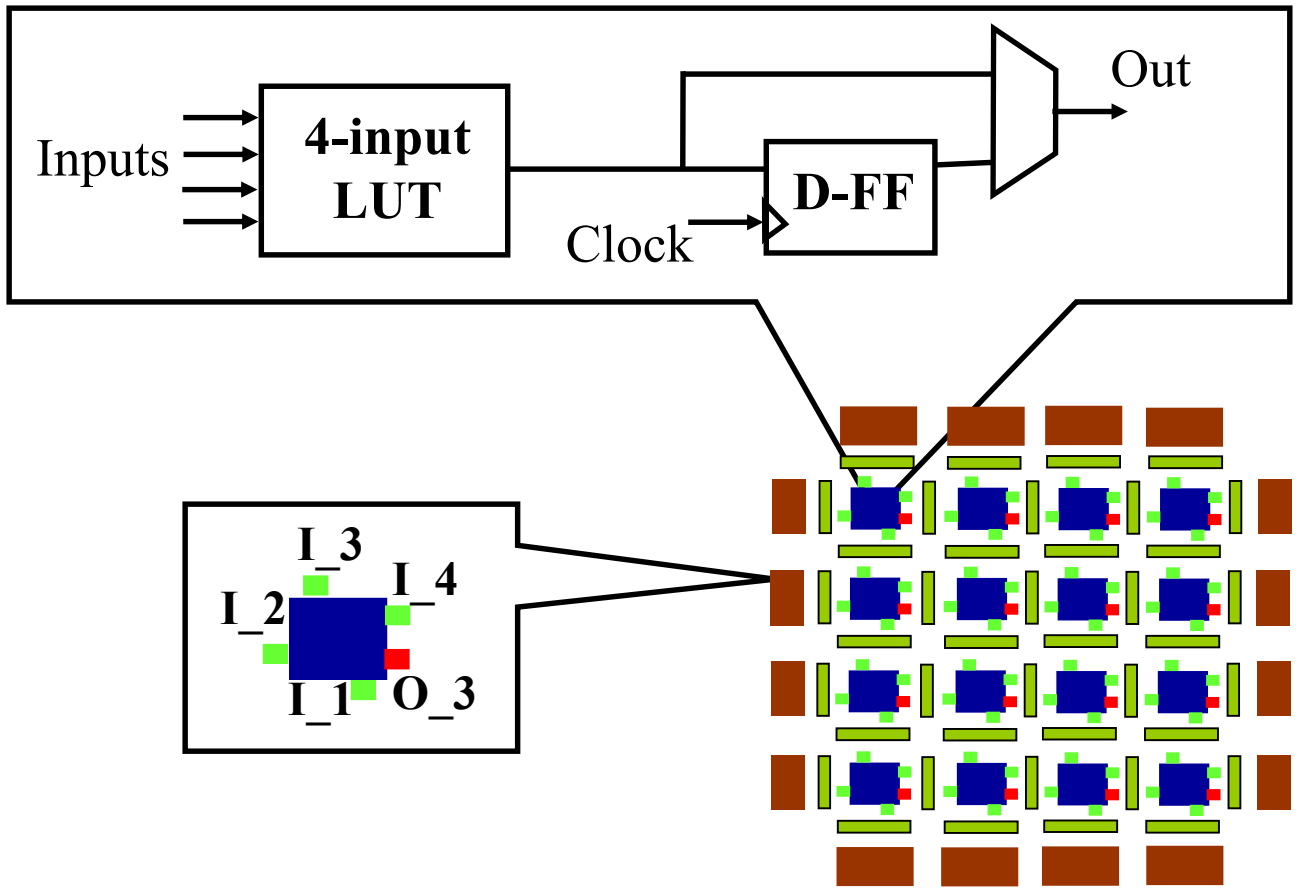
What is an FPGA?

- FPGAs(Field Programmable Gate Array) are pre-fabricated digital IC chips with programmable logic and programmable routing
  - A very popular implementation medium for digital systems
  - Can implement almost any digital design within its capacity
- FPGAs have the following key elements:
  - Basic logic cells
  - I/O logic cells
  - Programmable interconnect
  - Other resources like memory, clock distribution networks, etc

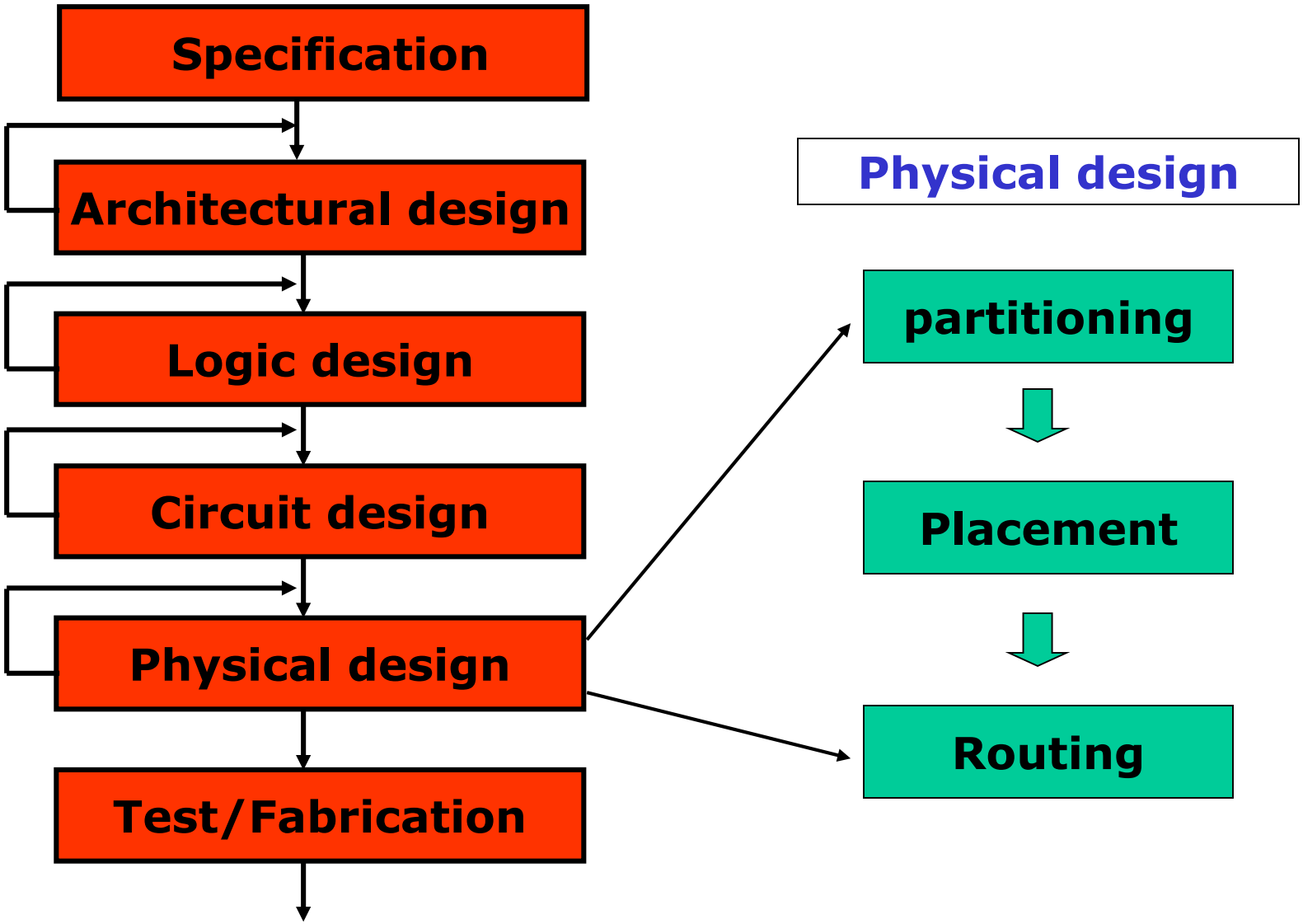
Island Style FPGA architecture



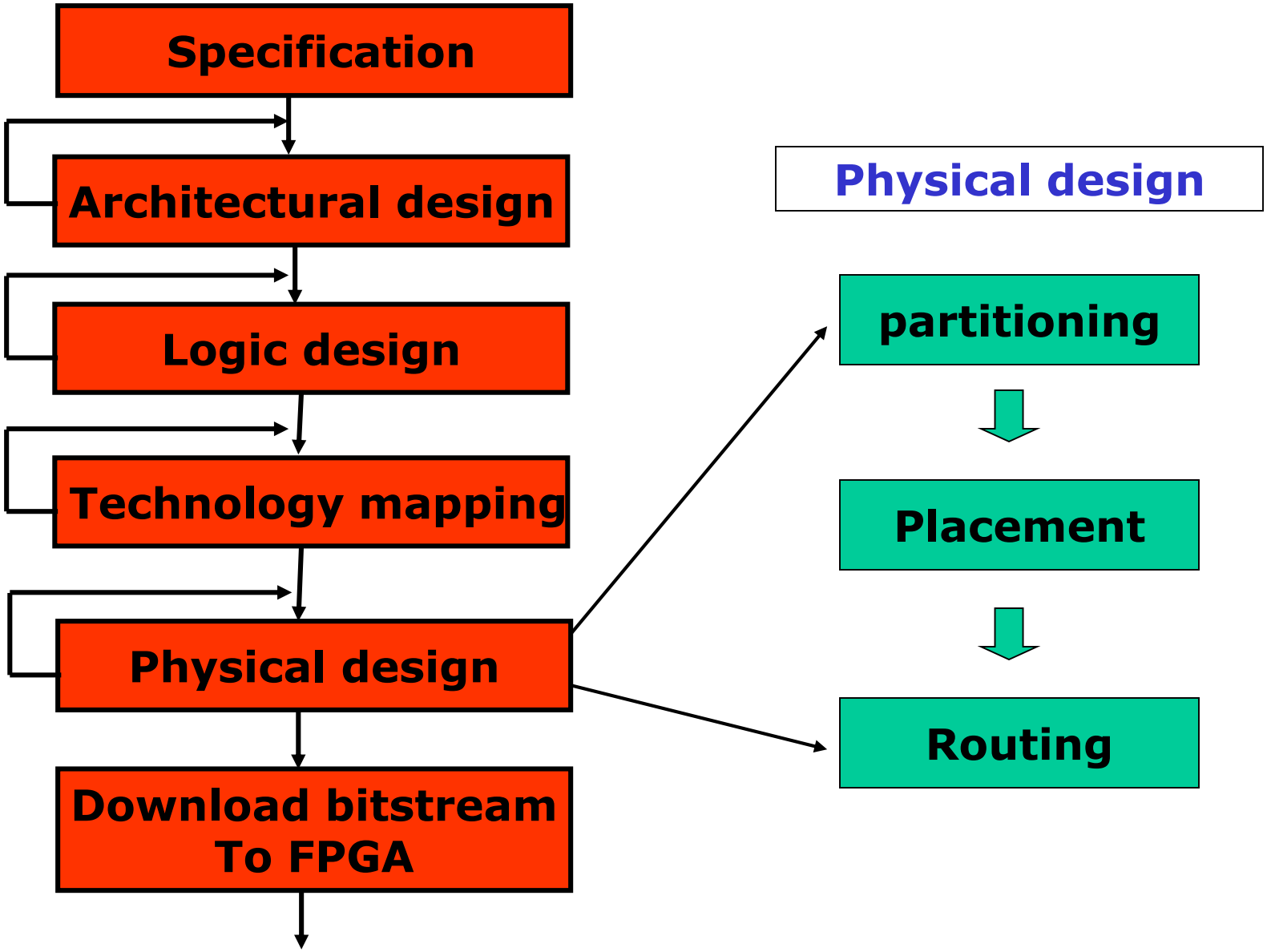
An abstract logic block



IC design flow



FPGA design flow



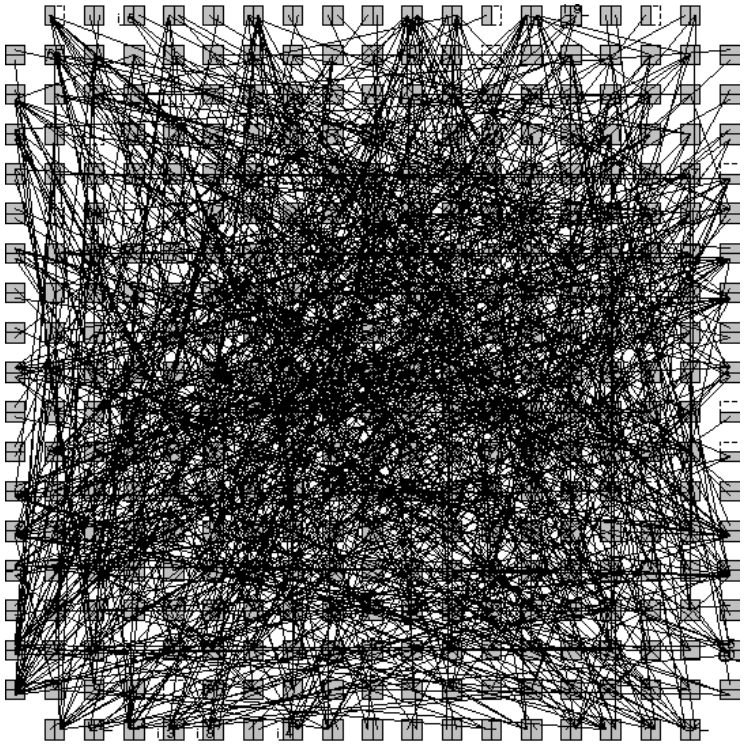
Placement definition

- Input(circuit design)
  - logic blocks
  - I/O pads
  - Inter connections
- Output:
  - Coordinates (x_i , y_i) for each block
- Objective:
  - The total wire length is minimized.
  - Also need to consider circuit delay and routability.
  - Other possible constraints such as pin locations

An example of placement

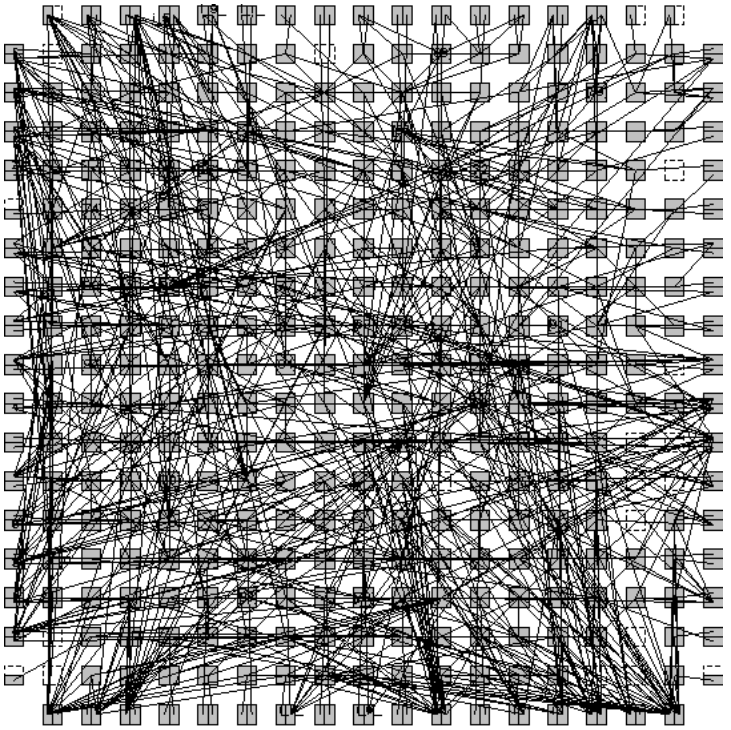
- MCNC Benchmark circuit e64 (contains 230 4-LUT). Placed on an FPGA

Random placement



Initial Placement. Cost: 74.5562. Channel Factor: 100

Final placement



Final Placement. Cost: 28.5384. Channel Factor: 100

General placement algorithms

- Simulated Annealing Placement
  - Initial placement. improve it by iterative swaps and moves
  - Example FPGA placement tool: VPR(1997)
- Partitioning-Based Placement
  - Recursively apply min-cut partitioning to map the circuit into the layout region
  - Example FPGA placement tool: PPFF(2003)
- Quadratic Placement
  - Build linear equations to minimize the squared wire length
  - To our knowledge, no FPGA placement tool developed based on Quadratic technique
- Hybrid and Hierarchical Placement
  - Example FPGA placement tool: Ultra-fast placement(1999)

Quadratic technique(1)

- Use squared wire length as objective function
- Quadratic placement builds linear equations to minimize the objective function
- Get the coordinates of logic blocks by solving linear equations
- Expand the design to entire chip area

Quadratic technique(2)

- Objective function

$$\Phi(x,y)=\frac{1}{2}\sum_{i,j}W_{ij}[(x_i-x_j)^2+(y_i-y_j)^2]=\frac{1}{2}\sum_{i,j}W_{ij}[x_i^2+x_j^2-2x_ix_j+y_i^2+y_j^2-2y_iy_j]$$

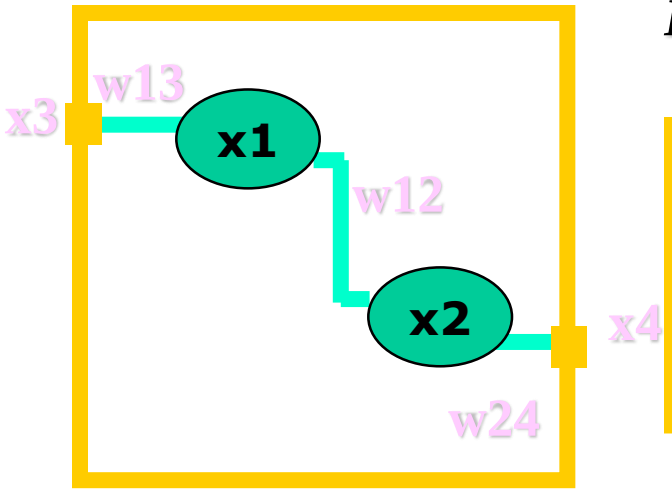
- In matrix form, for x direction only

$$\Phi(x,y)=\frac{1}{2}x^TQx+d_x^Tx+\frac{1}{2}y^TQy+d_y^Ty+const$$

- Matrix Q = [q_{ij}], vector d = [d_i]
  - W_{ij} contributes to q_{ii} and q_{jj}
  - -2 W_{ij} contributes to q_{ij}
  - d_i is the weight between node i and all fixed nodes

An example

$$\textit{Min: } 1/2[w_{13}(x1-x3)^2 + w_{13}(y1-y3)^2 + w_{12}(x1-x2)^2 + w_{12}(y1-y2)^2 + w_{24}(x2-x4)^2 + w_{24}(y2-y4)^2]$$



$$\textit{F: } 1/2[w_{13}(x1-x3)^2 + w_{12}(x1-x2)^2 + w_{24}(x2-x4)^2]$$

$$\partial F/\partial x1 = 0;$$

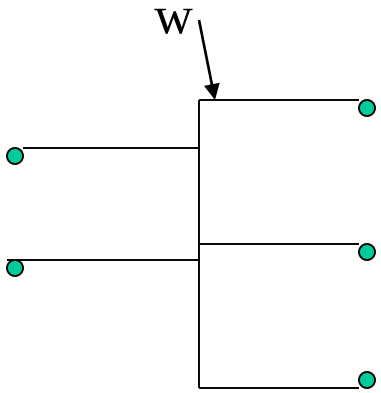
$$\partial F/\partial x2 = 0;$$

$$Qx = d$$

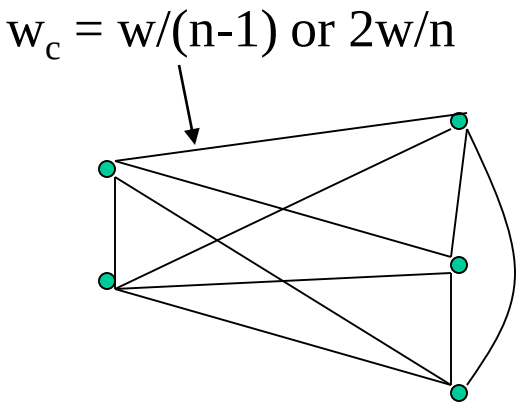
$$Q = \begin{bmatrix} w_{12}+w_{13} & -w_{12} \\ -w_{12} & w_{12}+w_{24} \end{bmatrix} \quad d = \begin{bmatrix} w_{13}x3 \\ w_{24}x4 \end{bmatrix} \quad x = \begin{bmatrix} x1 \\ x2 \end{bmatrix}$$

Net model

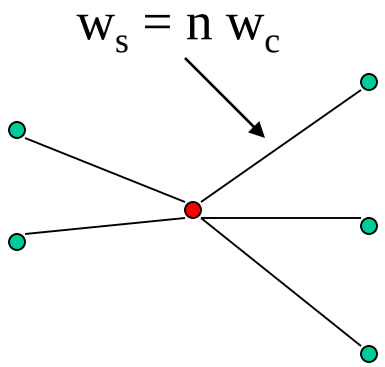
- Real circuits contain multi-terminal nets. They can not be used directly to build linear equations
- There are two net models to transform these nets to 2-terminal nets: clique model and star model
- Theses two net models are equivalent[3] if net weights are set appropriately



Multi-terminal net



Clique model



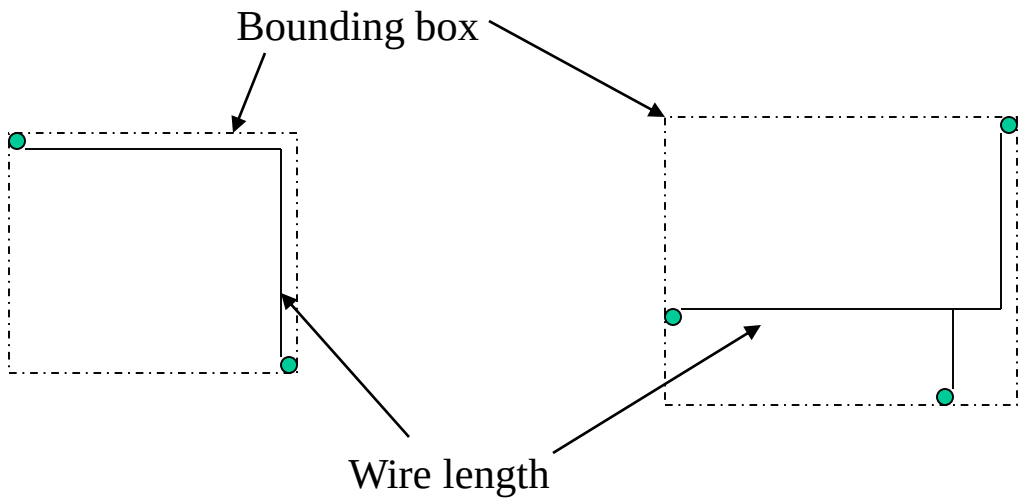
Star model

Outline

- Introduction to Placement for FPGAs
  - What is FPGAs?
  - What is Placement?
  - General placement algorithms
  - Quadratic placement algorithm
- Proposed quadratic placement algorithm
  - Problems with quadratic placement
  - proposed improvement
  - Details of proposed quadratic placement algorithm
- Experiment results
  - Benchmark circuits
  - Results and analysis
- Contribution
- Questions?

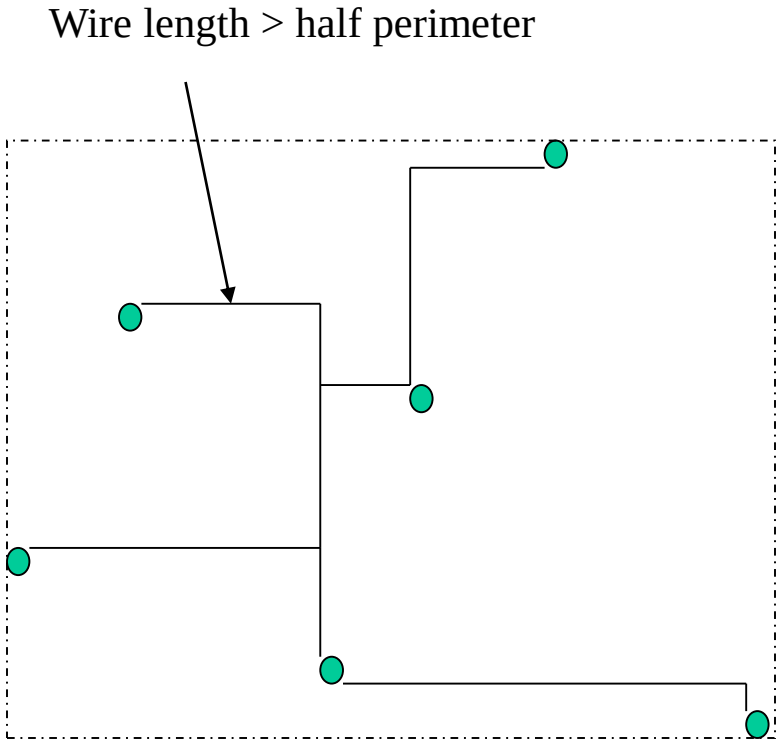
Half perimeter wire length

- We can not get accurate wire length until routing is performed
- Estimated wire length is used in placement tools
- Half perimeter wire length of bounding box is a good approach



Compensation for multi-terminal nets

- Bounding box wire length model usually underestimates the wire length to connect nets with more than 3 terminals[6]
- Factor q(i) is used to compensate it
- q(i)= 1 for nets with 3 or less terminals
- q(i) slowly increase to 2.79 for nets with 50 terminals
- q(i) = 2.79 +0.02616(terminals-50) for nets with more than 50 terminals

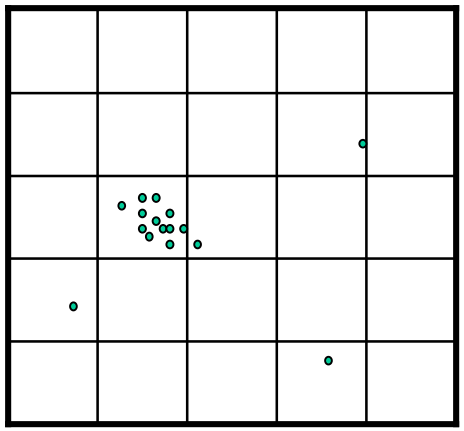


Compensation factor for multi-terminal nets

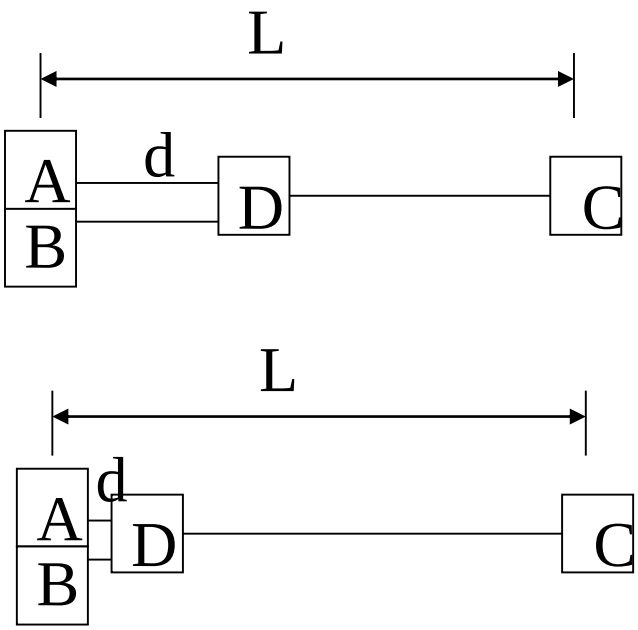
1.0, 1.0, 1.0, 1.0828, 1.1536, 1.2206, 1.2823, 1.3385, 1.3991, 1.4493,
1.4974, 1.5455, 1.5937, 1.6418, 1.6899, 1.7304, 1.7709, 1.8114, 1.8519, 1.8924,
1.9288, 1.9652, 2.0015, 2.0379, 2.0743, 2.1061, 2.1379, 2.1698, 2.2016, 2.2334,
2.2646, 2.2958, 2.3271, 2.3583, 2.3895, 2.4187, 2.4479, 2.4772, 2.5064, 2.5356,
2.5610, 2.5864, 2.6117, 2.6371, 2.6625, 2.6887, 2.7148, 2.7410, 2.7671, 2.7933

Problems of quadratic placement

- Problem 1: All nodes are in the center area
  - People use different methods to pull them apart



- Problem 2: Squared wire length is not always a good approach to linear wire length
  - Quadratic technique only will not achieve good placement



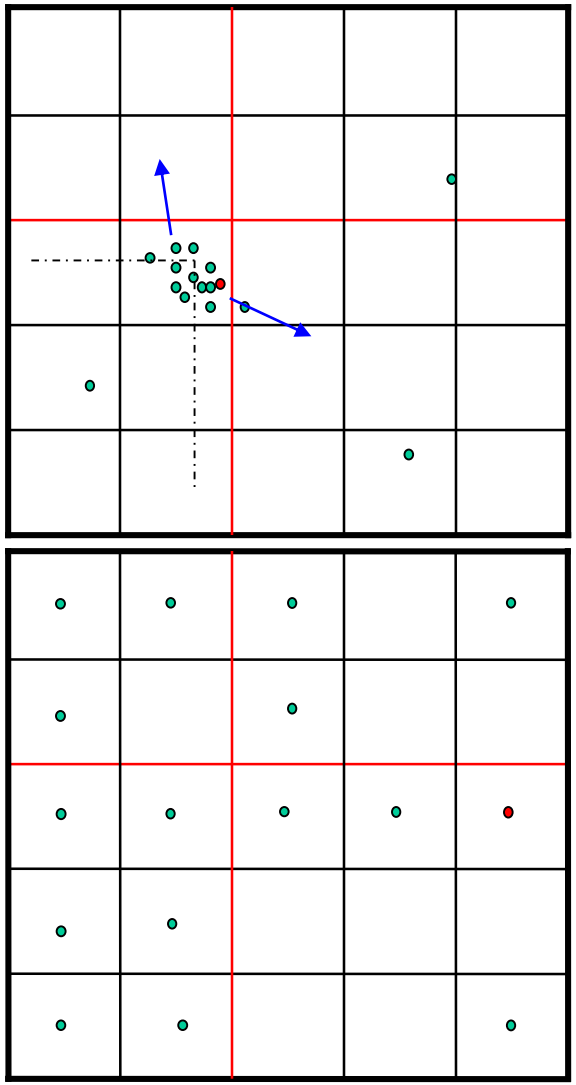
Solution to problems

- Problem 1: All nodes are in the center area
- Solution:
  - Based on current information ,find the target position for every node
  - Pull all nodes to their target position while maintaining the minimization of squared wire length

- Problem 2: Squared wire length is not always a good approach to linear wire length
- Solution:
  - For each node, find a direction to reduce the linear wire length
  - Move the node to its new position

How to find new position

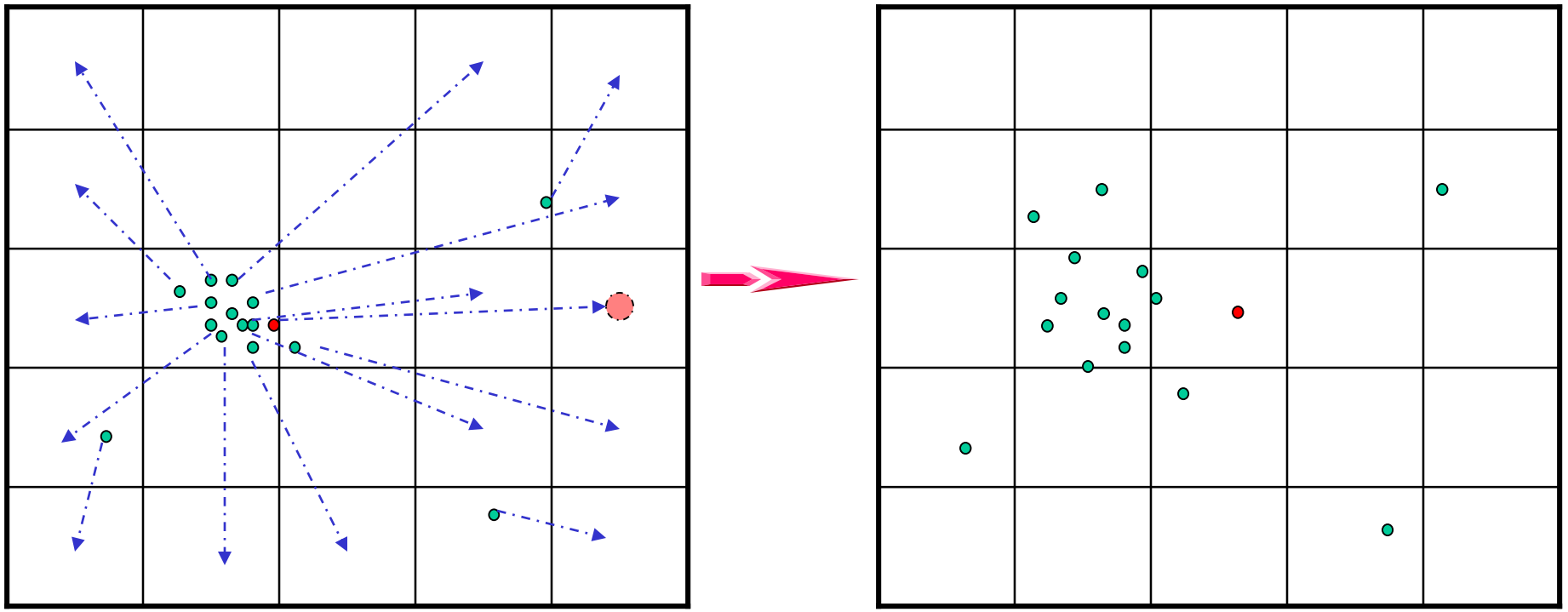
- Mapping the overlapped nodes into entire chip area according to their relative position
  - The entire chip area is divided into 4 parts
  - Move nodes to adjacent parts if any parts overflow
  - Recursively perform this process until every part is small enough(<4)
  - Place the nodes in local area according to their relative position



Reference placement

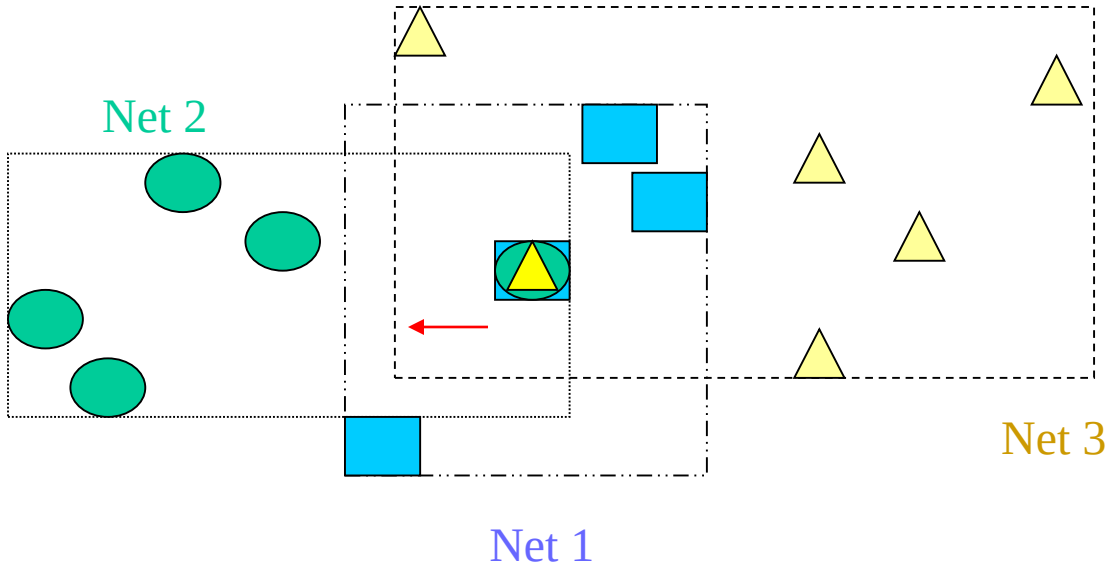
How to pull nodes apart

- Use the new position as in reference placement
- Add a dummy node for each node to pull the node to reference position
- The strength is determined by the average weight on the node
- Modify the equation and solve it again



How to find the direction

- If the node is the edge of a net, then it is possible to reduce the half perimeter wire length by moving it toward the center of the net
- If it is the edges of more than one nets, check the number of terminal of the nets, move it to the direction with larger q(i)
- If it is not the edge of a net, it is free to move within the bounding box
- Perform it in both x and y direction



*nodes can move in both x and y directions

How to move the nodes

- Find the target position that minimizes the linear wire length
- Adding dummy nodes and re-solve the equation
  - When apply this together wit h problem 1, do it the same as problem one
- Modify the coordinates directly
  - When apply this to the final placement. Move the node directly to the target position

Proposed algorithm

Stage 1

- Quadratic equation solver
- Add dummy nodes and expand the placement
- Linear wire length refinement
- Repeat until there is no significant improvement

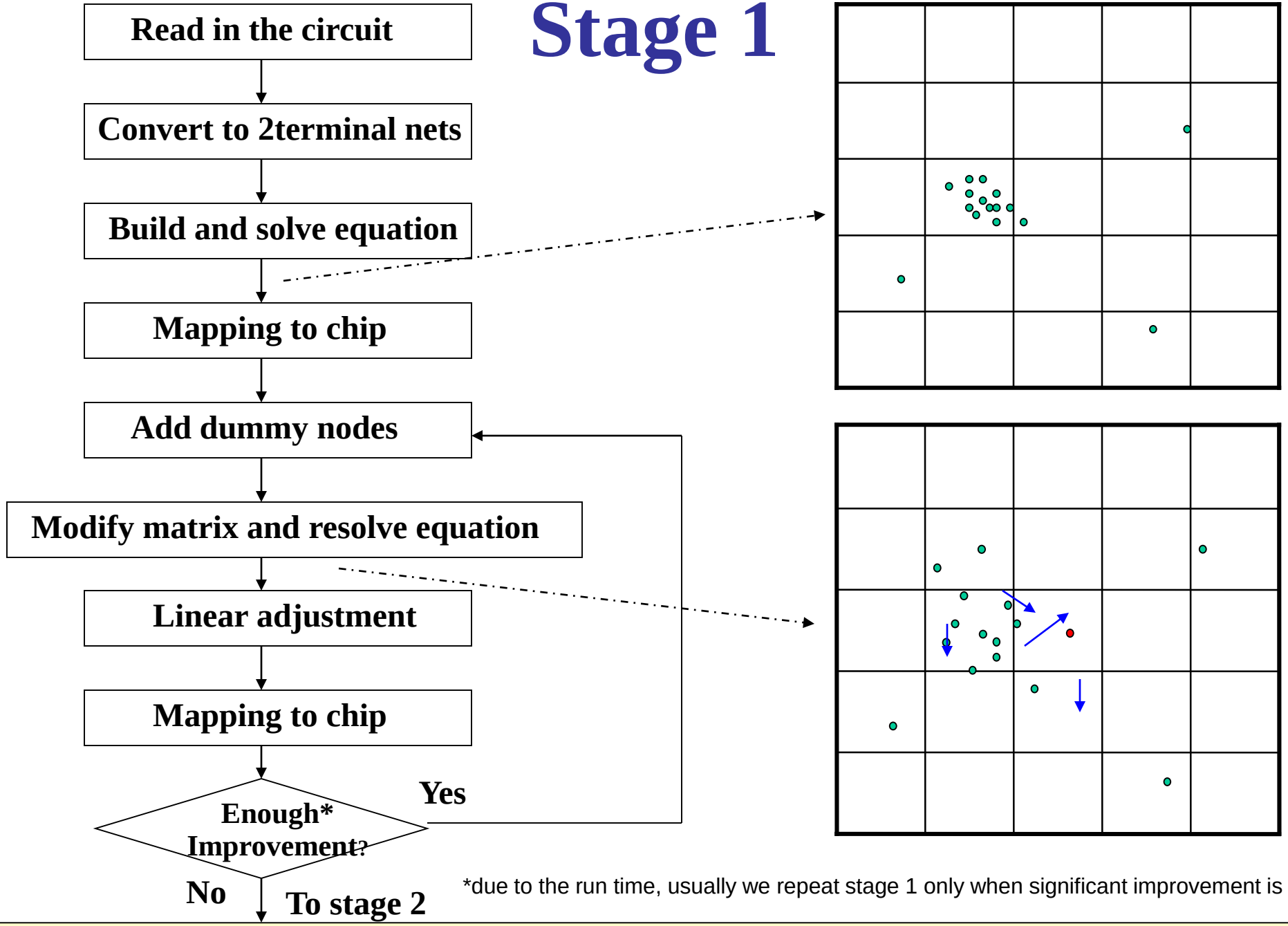
Stage 2

- Linear wire length refinement on legal placement
- Repeat until there is no more improvement

Stage 3

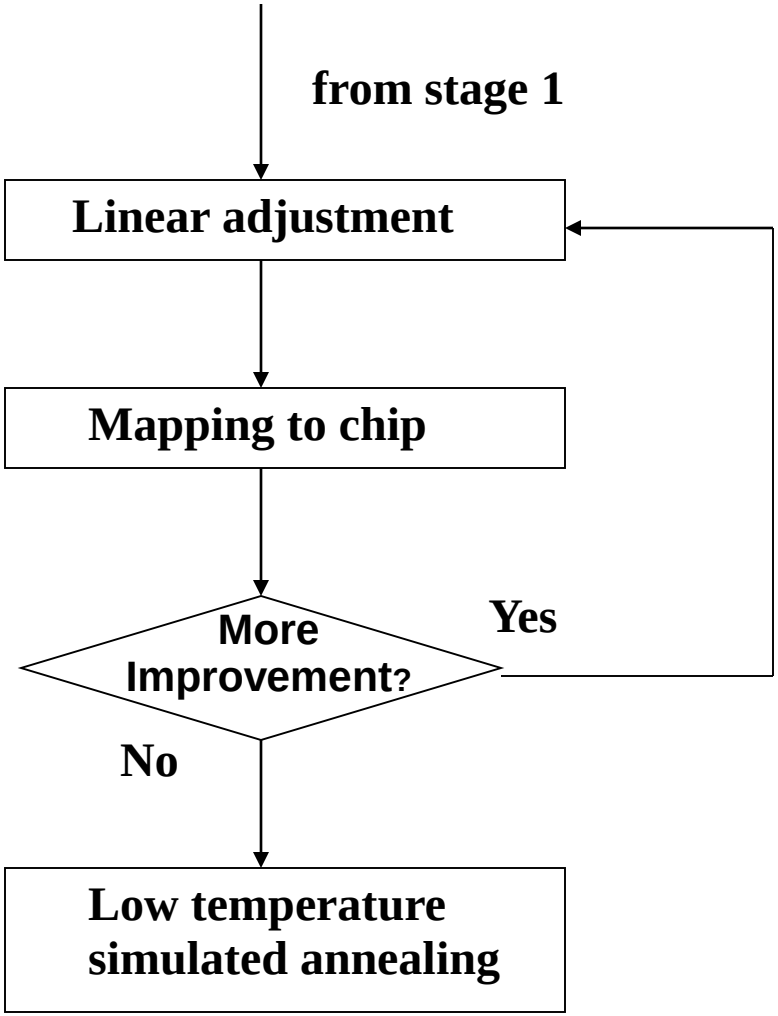
- Low temperature simulated annealing refinement

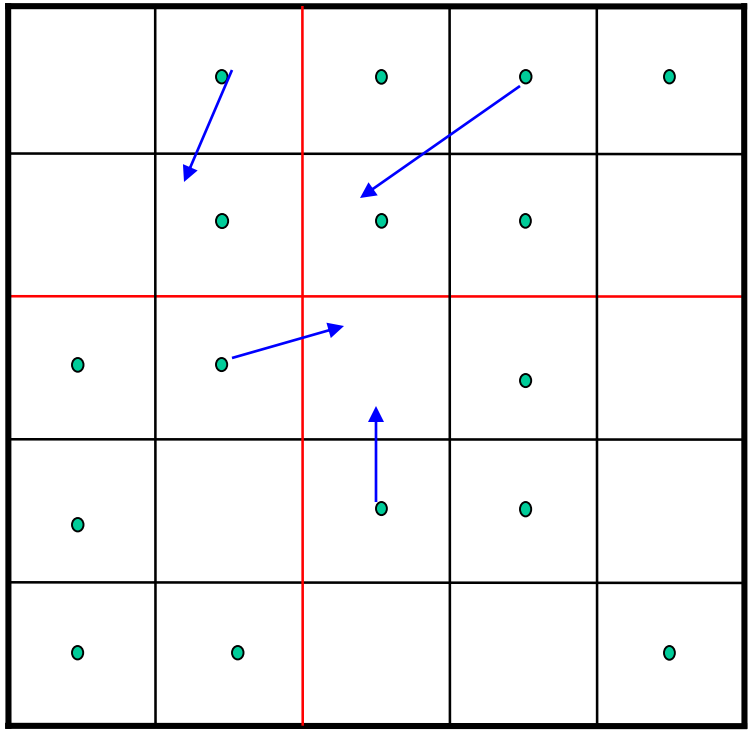
Stage 1



*due to the run time, usually we repeat stage 1 only when significant improvement is achieved

Stage 2, 3





*The linear adjustment in stage 2 is different from that in stage one. We do not modify and resolve equations in this stage

Outline

- Introduction to Placement for FPGAs
  - What is FPGAs?
  - What is Placement?
  - General placement algorithms
  - Quadratic placement algorithm
- Proposed quadratic placement algorithm
  - Problems with quadratic placement
  - proposed improvement
  - Details of proposed quadratic placement algorithm
- Experiment results
  - Benchmark circuits
  - Results and analysis
- Contribution
- Questions?

- We compare our results with VPR
- VPR is state of the art FPGA placement tool. Current Altera FPGA tools are based on it.
- Altera is a leading FPGA company
  - VPR is based on simulated annealing algorithm
    - Initial placement improved by iterative swaps and moves
    - Accept swaps/moves if they improve the cost
    - Accept bad swaps/moves to avoid being trapped in local minimum
    - Parameter T is used to control the moves/swaps
  - It uses adaptive annealing schedule
    - Cost Function
    - Temperature Update
    - Stopping Criteria

Benchmark circuits

No	name	#blk	#net	FPGA area
1	alu4	1544	1536	40*40
2	Apex2	1919	1916	44*44
3	Apex4	1290	1271	36*36
4	Bigkey	2133	1936	54*54
5	Clma	8527	8445	92*92
6	Des	2092	1847	63*63
7	Diffeq	1600	1561	39*39
8	Dsip	1796	1599	54*54
9	Elliptic	3849	3735	61*61
10	Ex1010	4618	4608	68*68

No	name	#blk	#net	FPGA area
11	Ex5p	1135	1072	33*33
12	Frisc	3692	3576	60*60
13	Misex3	1425	1411	38*38
14	Pdc	4631	4591	68*68
15	S298	1941	1935	44*44
16	S38417	6541	6435	81*81
17	S38584	6789	6485	81*81
18	Seq	1826	1791	42*42
19	Spla	3752	3706	61*61
20	tseng	1221	1099	33*33

These are 20 MCNC benchmark circuits used by vpr. They are transformed to .net format

FPGA area is the FPGA size required to place the circuit. E.g. 40x40 array of CLBs is required for circuit alu4

Experiment results:wire length

No	name	Q	VPR	rate
1	alu4	198.34	193.56	1.025
2	Apex2	276.80	273.41	1.012
3	Apex4	186.52	183.48	1.017
4	Bigkey	318.37	314.49	1.012
5	Clma	1517.9	1466.9	1.035
6	Des	392.19	388.37	1.010
7	Diffeq	171.70	169.04	1.016
8	Dsip	307.03	304.02	1.010
9	Elliptic	558.62	546.36	1.022
10	Ex1010	669.22	657.76	1.017

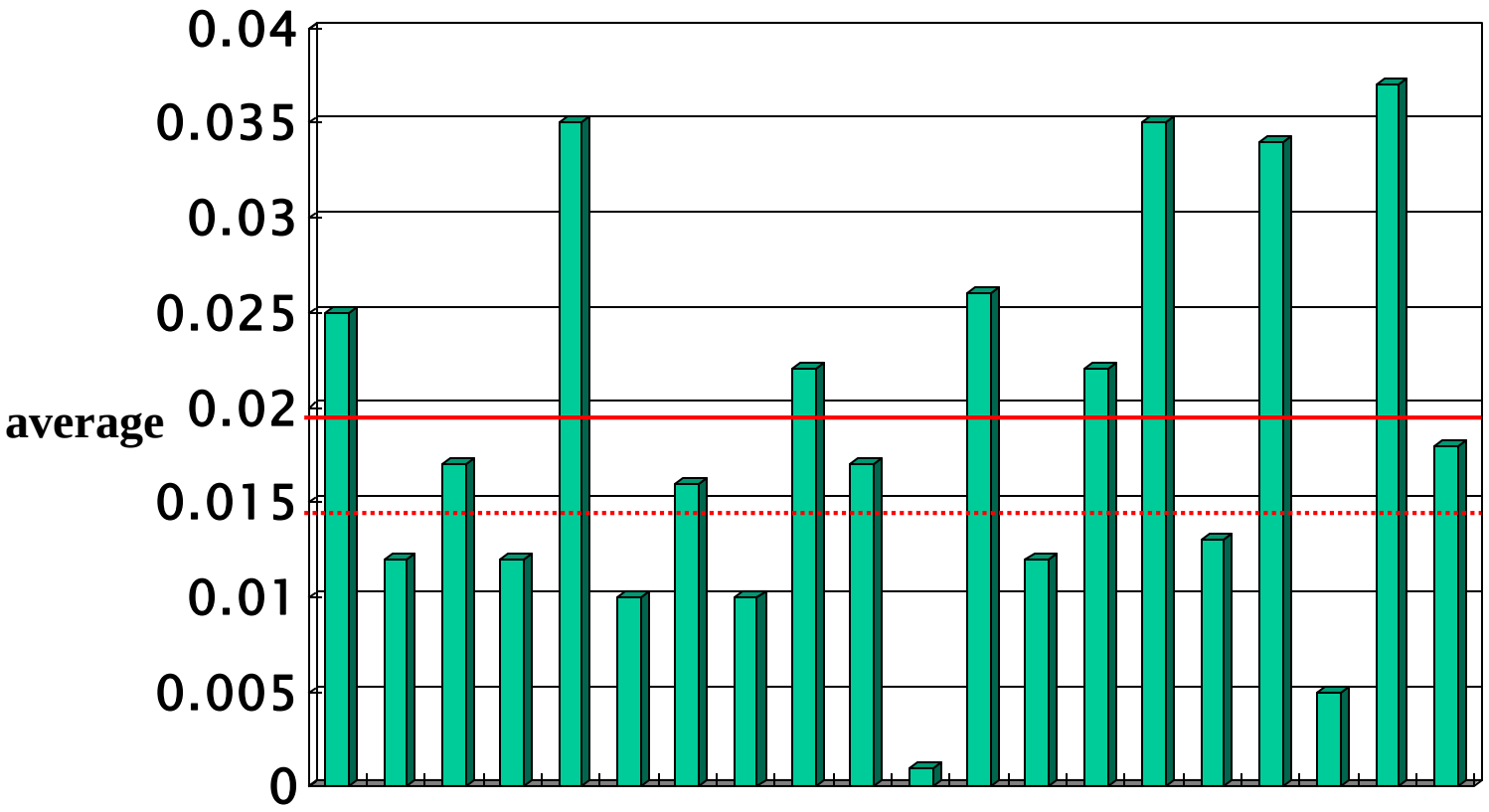
No	name	Q	VPR	rate
11	Ex5p	174.00	173.82	1.001
12	Frisc	564.54	550.30	1.026
13	Misex3	193.89	191.65	1.012
14	Pdc	923.39	903.38	1.022
15	S298	211.15	204.04	1.035
16	S38417	709.52	700.41	1.013
17	S38584	829.92	802.65	1.034
18	Seq	266.01	264.77	1.005
19	Spla	647.69	624.40	1.037
20	tseng	117.34	115.3	1.018

Average rate is 1.019

VPR runs in wire length driven mode

The total wire length rate after routing is 1.014

Wire length comparison



Wire length penalty

Experiment results:run time

No	name	Q	VPR	rate
1	alu4	6	33	5.500
2	Apex2	6	49	8.167
3	Apex4	5	27	5.400
4	Bigkey	9	51	5.667
5	Clma	190	1123	5.911
6	Des	6	46	7.667
7	Diffeq	7	37	5.286
8	Dsip	9	36	4.000
9	Elliptic	56	163	2.911
10	Ex1010	46	290	6.304

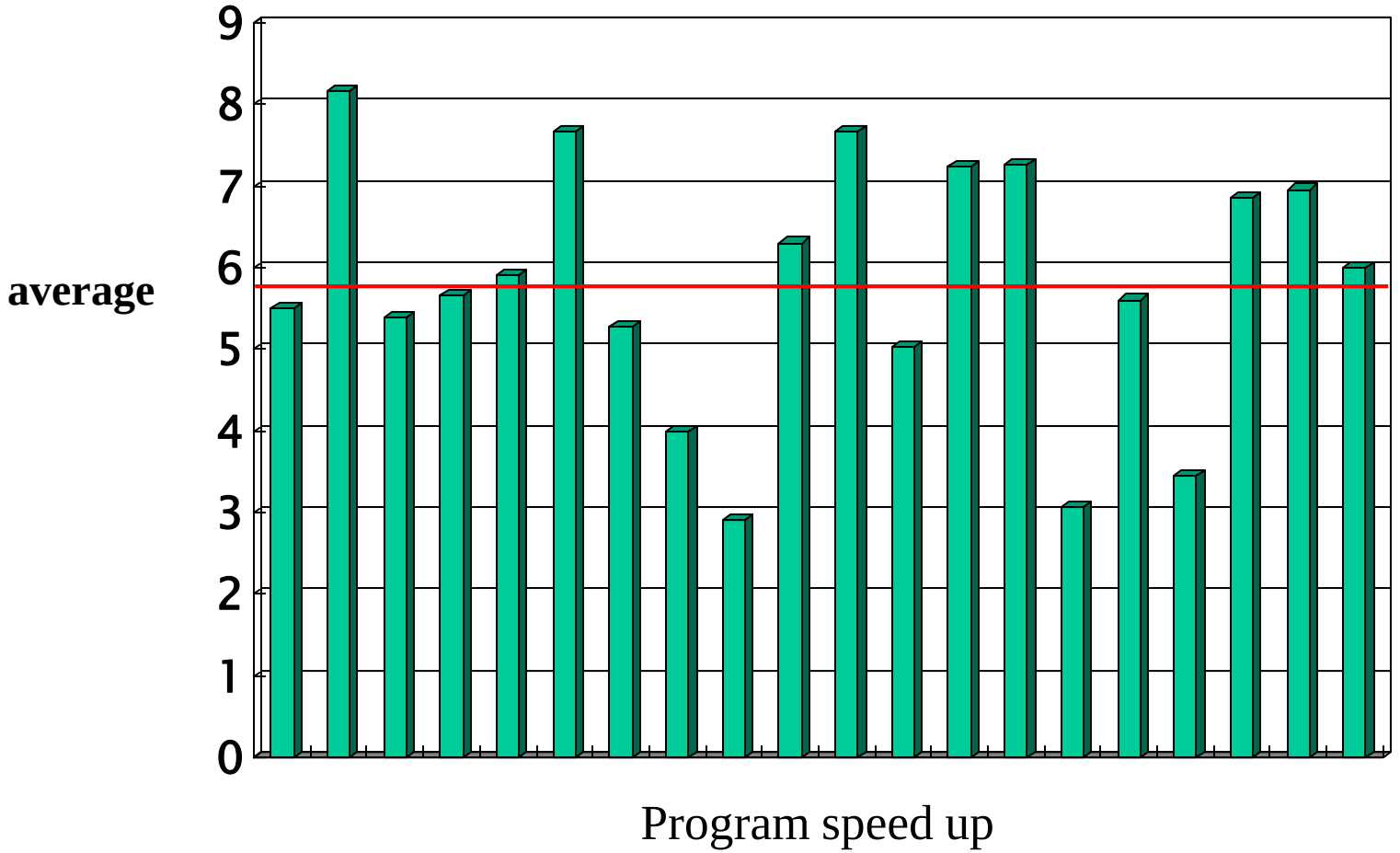
No	name	Q	VPR	rate
11	Ex5p	3	23	7.667
12	Frisc	34	171	5.029
13	Misex3	4	29	7.250
14	Pdc	42	305	7.262
15	S298	14	43	3.071
16	S38417	129	723	5.605
17	S38584	220	758	3.445
18	Seq	7	48	6.857
19	Spla	25	174	6.960
20	tseng	4	24	6.000

Average rate is 5.798

VPR runs in wire length driven mode

Time is measured in second

Run time comparison



Experiment results

- We run VPR in wire length driven mode
- The quality is almost the same as VPR
  - Estimated wire length is less than 2% longer than VPR
  - We routed our placement and compared with VPR. The actual wire length after routing is only 1.4% longer than VPR
- Run time is 5.8x faster than VPR
  - 8x times faster in the best case

Contributions

- Quadratic placement technique is very efficient and popular in real world placement tools for IC design
  - To our knowledge our work is the first investigation of quadratic placement technique for FPGAs
- The quality is almost same as VPR
  - Estimated wire length is less than 2% longer than VPR
  - The actual wire length after routing is only 1.4% longer than VPR
- Run time is 5.8x faster than VPR
- Research paper in preparation for submission to FPL'05
  - A well known annual International Conference on Field Programmable Logic and Applications

References

- 1. A. Marquardt, V. Betz, and J. Jose, “Timing-Driven Placement for FPGAs”, FPGA’ 2000
- 2. P. Maidee, C. Ababei and K Bazargan, “Fast Timing-driven Partitioning-based Placement for Island Style FPGAs”, DAC’2003
- 3. N.Viswanathan, and C.Chu, “FastPlace: Efficient Analytical Placement using Cell Shifting, Iterative Local Refinement and a Hybrid Net Model”, ISPD’04
- 4. Y. Sankar, J. Jose, “Trading Quality for Compile Time: Ultra-Fast Placement for FPGAs” , FPGA’ 1999
- 5. C. J. Alpert, T. F. Chan, D. J. H. Huang, I. L. Markov and K. Yan, “Quadratic Placement Revisited”, DAC’1997
- 6. C. Cheng, “RISA: Accurate and Efficient Placement Routability Modeling”, ICCAD, 1994
- 7. K. Shahookar and P. Mazumder, “VLSI Cell Placement Techniques,” ACM Computing Surveys, vol. 23, no. 2, June 1991, pp. 143-220

Questions?